Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

CRT-2005-21

EVE-OPT: MEMETIC ALGORITHM FOR THE CAPABILITY VEHICLE ROUTING PROBLEM

by

Simone Palamara
Guido Perboli
Roberto Tadei

June 2005

Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7 Canada
Téléphone : (514) 343-7575 / Télécopieur : (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

HEC ÉCOLE POLYTECHNIQUE MONTRÉAL Université de Montréal

PUBLICATION
CRT-2005-21

EVE OPT: MEMETIC ALGORITHM FOR THE CAPABILITY VEHICLE ROUTING PROBLEM

by

Simone Palamara [1]
Guido Perboli [1,2]
Roberto Tadei [1]

This research was supported in part by the Discovery Grants program of the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the HP Invent Philanthropic Grants.

[1] Politecnico di Torino, Dipartimento di Automatica ed Informatica, C.so Duca degli Abruzzi 24, 10129 Torino, Italy (simone.palamara@polito.it, guido.perboli@polito.it, roberto.tadei@polito.it)

[2] Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2005

105-06990/

Abstract

In this paper, we present *EVE-OPT*, a Memetic Algorithm based on Genetic Algorithms and Taboo Search for solving the Capability Vehicle Routing Problem. Several Memetic Algorithms have been proposed in the last years for solving this problem. Despite their good results, they usually use strongly problem dependent neighborhoods and complex genetic operators, that make difficult their application to real problems, where several additional constraints must be considered. Our Memetic Algorithm hybridizes two very simple heuristics, introduces a new genetic operator, the *Chain Mutation*, as well as a new mutation scheme. We also apply a procedure, the *k-chain-moves*, able to increase the neighborhood size, improving the solution quality, with a negligible computational effort. Despite its simplicity, *EVE-OPT* is able to achieve the same results of more complex state-of-the-art algorithms.

Keywords: Vehicle Routing, Memetic Algorithm, Genetic Algorithm, Taboo Search.

Résumé

Cet article décrit *EVE-OPT*, une heuristique mémétique qui combine un algorithme génétique et un algorithme tabou pour la confection des tournées de véhicules. Beaucoup d'algorithmes mémétiques ont été proposés pour ce problème dernièrement. Malgré la bonne qualité des solutions obtenues, ils utilisent des opérateurs génétiques complexes et des voisinages qui dépendent étroitement du problème. Ceci complique l'utilisation de ces algorithmes dans les problèmes réels, où l'on doit tenir compte de plusieurs contraintes additionnelles. Notre algorithme mémétique fusionne deux heuristiques très simples, il introduit un nouvel opérateur génétique, *Chain Mutation*, et un nouveau schéma de mutation. Nous utilisons aussi une procédure, appelée *k-chain-moves*, qui permet d'augmenter la taille du voisinage et la qualité des solutions avec un effort de calcul réduit. Malgré sa simplicité, *EVE-OPT* peut obtenir des résultats comparables aux autres algorithmes.

Mots clés: Confection des tournées de véhicules, algorithme mémétique, algorithme génétique, tabou.

1 Introduction

Transportation and Routing costs may affect up to 20% of the final costs. For this reason Vehicle Routing Problems (VRPs) have been intensively studied in the past decades and constitute a large field in Combinatorial Optimization research [31]. Given a depot, a set of customers with a given non-negative demand and location, and a set of vehicles with equal capacity, the Capacitated Vehicle Routing Problem (CVRP) consists in finding a collection of exactly K circuits (each corresponding to a vehicle route) with minimum cost, defined as the sum of the costs of the arcs belonging to the circuits, and such that:

- each circuit visits the depot;
- each customer is visited by exactly one circuit;
- the sum of the demands of the customers visited by a circuit does not exceed the vehicle capacity.

Moreover, the costs matrix satisfies the triangle inequality:

$$c_{ik} + c_{kj} \leq c_{ij} \quad \forall i, j, k \in V \tag{1}$$

(See Figure 1 for an example of solution).

Several approaches as Simulated Annealing, Taboo Search, Genetic Algorithms, Ant Systems have been developed in the past decades to solve the CVRP. In the last two years the research has been focused on Memetic Algorithms (MAs) [9]. Unfortunately, the best MAs use complex and strongly problem dependent genetic operators and neighborhoods, making difficult to introduce additional constraints to the original problem and to adapt the developed methods to real applications.

In this paper we present Evolutionary VEhicle OPTimization (EVE-OPT), a Memetic Algorithm for solving the CVRP combining Genetic Algorithm (GA) and Taboo Search (TS). The ingredients of our GA are a natural representation of the solutions and a recombination able to transmit meaningful characteristics from parents to children. To introduce new genetic material and maintain genetic diversity, parent solutions are combined with a crossover operator called Partially Mapped Crossover (PMX). Furthermore, we introduce a new kind of mutation operator, the Chain Mutation, able to explore new regions of the solution space and to escape from local optima. Each solution generated by the GA is further optimized by the TS. Our TS algorithm combines two different neighborhoods and a procedure able to enhance neighborhood size without increasing its complexity called k-chain-moves, originally introduced by Crainic, Perboli and Tadei for the Multi-dimensional Bin Packing Problem (.[22], [10]).

The remainder of this paper is organized as follows. In Section 2 are presented some of the best metaheuristics proposed in recent years for the Vehicle Routing Problem. In Section 3 the main features of GAs and TS are discussed. Section 4 is devoted to present our algorithm, whose computational tests are discussed in

Section 5. Finally, in the last section are presented the conclusions and the future developments for this project.

2 State of the art

Several industrial procedures can be reducted to the CVRP. For this reason, many algorithms have been developed in the last years. These algorithms can be classified in the following three classes:

- *Local Search*: it includes Descent Algorithms (Ergun, Orlin and Steele-Feldman [11]), Simulated Annealing (Osman [21]), Deterministic Annealing (Golden, Wasil, Kelly and Chao [16]; Li, Golden and Wasil [17]), Taboo Search (Osman [21]; Taillard; Gendreau, Hertz and Laporte [12]; Xu and Kelly [32]; Rego and Roucairol [25]; Rego [24]; Barbarosoğlu and Ögür [3]; Cordeau, Laporte and Mercier [8], Toth and Vigo [30]).
- *Learning Mechanism*: it includes Neural Networks (Ghaziri [13, 14]; Matsuyama [18]; Schumann and Retzko [28]) and Ant Algorithms (Reimann, Doerner and Hartl [26]).
- *Population Search*: it includes Evolutionary Algorithms (Prins [23]; Berger and Barkaoui [5]) and Adaptive Memory Procedures (Rochat and Taillard [27]; Tarantilis and Kiranoudis [29]).

In the following, for each class we will present the best algorithms proposed in the last years. For further details, it is possible refer to the last survey by J.F. Cordeau, M. Gendreau, A. Hertz, G. Laporte and J.S. Sormany [9].

2.1 Local Search

Cordeau, Laporte and Mercier (2001) proposed a TS algorithm, called *Unified Taboo Search Algorithm* (UTSA) [8], to solve periodic and multi-depot VRPs. It tolerates intermediate unfeasible solutions through the use of a generalized objective function containing self-adjusting coefficients. This feature allows to decrease the average deviation from the best known solution without any further computational effort.

The *Granular Taboo Search* (GTS) [30], by Toth and Vigo (2003), is based on the main idea that removing all nodes unlikely to appear in an optimal solution could considerably reduce the neighborhood size and thus computational time.

In 2003 Ergun, Orlin and Steele-Feldman proposed a descent mechanism, the *Very Large Neighborhood Search* (VLNS) [11], trying to pursue iteratively for an improving solution by exploring a very large neighborhood with respect to the input data. The main advantage is that a broad search can be performed considering several moves at once but, on the other side, it requires a considerable computational effort.

Li, Golden and Wasil in their VRTR (2004) [17] combines the record-to-record principle with a variable-length neighbor list whose principle is inspired to GTS. When there is no improvement for a fixed number of iterations, a final attempt is made to improve the best known solution by means of a perturbation technique.

2.2 Learning Mechanism

The best algorithm in the Learning Mechanism class is the *D-ants heuristic* by Reimann, Doerner and Hartl (2004). It iteratively applies two phases until a stopping criterion is met. The first phase aims to find the best solution by applying a savings based procedure to generate a pool of good solutions and improve them. In the second phase, the best solution identified is decomposed into several subproblems, optimized by means of the same first phase procedure.

2.3 Population Search

Tarantilis and Kiranoudis (2002) developed an adaptive memory procedure that does not initiate new solutions by combining full vehicle routes, but route segments extracted from good quality routes [29].

Prins (2004) [23] combined crossover and mutation operators. Improvements are obtained by means of a Local Search procedure applied to a candidate solution that ends when the first improving move is met. This procedure is also applied after each mutation.

The *Active Guided Evolution Strategies* (AGES) algorithm of Mester and Bräysy (2004) [19] combines guided Local Search with evolution strategies into an iterative two-stage procedure. AGES concentrates on single parents, creating a single offspring from a parent through a mutation procedure. Offsprings presenting a better fitness value replace their parents. A restart mechanism applied to the best current solution covers a significant role in improving solutions quality.

The hybrid genetic algorithm by *Berger and Barkaoui* (2004) [5] is a Memetic Algorithm managing two different populations. Each population size is kept constant by replacing the worst elements by the best ones. Solutions are improved by performing a large scale neighborhood search that combines three sequential insertion mechanisms, and by applying a route improvement scheme. A final attempt to improve each route is made by removing in turn each customer and reinserting it according to an insertion mechanism.

3 Memetic Local Search

A Memetic Algorithm (MA) is a population-based approach for heuristic search in optimization problems. Basically, it combines Local Search heuristics with Evolutionary procedures. In particular, our MA is based on the joint use of Genetic Algorithm and Taboo Search. In the following a quick description of the two methods is given.

3.1 Genetic Algorithms

Genetic Algorithms (GAs) [20], introduced by John Holland in 1960s, are stochastic search techniques that imitate the mechanism of natural evolution. Unlike other search techniques, GAs consider multiple designs during the same iteration. Such a set of designs is referred to as a population. The designs in a population are called chromosomes or individuals, and are associated to a fitness function defined such that good designs have high fitness values. Moreover, each chromosome is constituted by genes. During each iteration, the individuals which constitute the population apply three types of operators in order to give origin to new chromosomes: selection, crossover and mutation. Selection is the operation that guides the algorithm to the solution by preferring individuals with high fitness over low-fitted ones. Crossover randomly splits and merges chromosomes, such that children inherit genes from both parents. Mutation alters the value of a few randomly selected genes throughout the population with a predetermined probability. The new solution is re-inserted in the population.

3.2 Taboo Search

Taboo Search (TS) is a metaheuristic algorithm developed by Glover [4]. It partially explores the solution space by means of the definition of moves. In order to avoid to be trapped into local optima, the TS implements a mechanism that forbids some selected moves. At each iteration, the move carried out is the most promising among those available. Formally, the main elements of the algorithm are:

- Neighborhood: a function mapping each feasible solution CS into a set of other solutions. Every time the algorithm has to consider a new solution chosen from the neighborhood of the current solution;
- Taboo List (TL): a list containing the last T moves carried out, which for this reason are forbidden. A solution obtained from the current solution CS with a move contained in the Taboo List, if it does not verify the aspiration criterion, can not be a member of the neighborhood of CS; it has been inserted to prevent the search becoming trapped in local minima;
- Aspiration criterion: if a Taboo move (move which is contained in the Taboo List) satisfies this criterion, then the solution obtained by applying it to the current solution CS can be considered to be in the neighborhood of CS. The usual criterion is that the move produces a solution that improves the best solution obtained so far.
- Stop criterion: this criterion verifies when the algorithm must stop.

4 Evolutionary VEhicle OPTimization (EVE-OPT)

In this section we describe in detail the features of our algorithm, referring particularly to their influence on the global behavior. The choice of combining an evolutionary procedure as GA with the TS is due to the need of overrunning mutual deficiencies. Indeed, the GA is able to explore a larger solutions space subset respect to the TS, but it is slower to converge to a good solution. All GA and TS features described in the following are properly defined to originate an algorithm as simple as possible, able to run without substantial alterations over instances with additional constraints, as in the real world applications. The algorithm scheme is presented in Figure 2. Our algorithm starts generating an initial population, then it selects chromosomes to be reproduced. They are combined through a crossover operator to generate an offspring, which is then modified by a mutation operator with a certain probability. The resulting offspring is further refined by the TS procedure. The new solution substitutes a population member chosen according to the ReInsertion Probability. This procedure is iterated until the stop criterion is not met.

4.1 Common features

In the following we present those features exploited both by the GA and the TS.

4.1.1 Solution Representation

in order to apply GAs to a particular problem we have to encode a generic solution of the problem into a chromosome. How to encode a solution is a key-issue for the success of GAs. Canonical GAs use binary encoding of individuals on fixed-length strings. Such a representation is not naturally suitable for ordering problems such as the Traveling Salesman Problem and the VRP, because no direct and efficient way has been found to map all the possible solutions into binary strings. Hence, concerning the representation, we use a fixed size chromosome. Possible values for genes are an integer representing a customer node or a special catcher able to separate different routes. If in a chromosome there are two or more consecutive catchers, they are interpreted as being a single separator between two routes. Since we don't know beforehand how many vehicles will be used in the optimal solution, we must include a safe number of catcher genes. We set this number to $4 * K_{min}$, where K_{min} is a lower bound of the minimum number of vehicles needed. It can be achieved in the following way:

$$K_{min} = \left\lceil \frac{\sum_{i=1}^{n} d_i}{C} \right\rceil \tag{2}$$

The customers are represented as integer values between 0 and $n-1$, where n is the number of the customers, while catchers value is n. We will resort to an example in order to better describe our proposal. The chromosome presented in Figure 6 codifies a solution equivalent to the one presented in Figure 1. Figure

6 shows a chromosome presenting five catchers able to codify solutions with 1 up to 6 vehicles (the catcher's value is 11. In the example there exist three vehicles, so three catchers are redundant. In fact, the second and the third vehicle are separated by two catchers and one of them is redundant, like the last two. The vehicle paths represented are $\{depot, 2, 4, 1, depot\}$, $\{depot, 7, 5, 8, 10, depot\}$ and $\{depot, 9, 6, 3, depot\}$.

4.2 GA features

4.2.1 Population

the initial population contains a fixed number of chromosomes which are computed using the heuristics by Clarke and Wright [7]. During all the evolutionary process the population size remains constant.

4.2.2 Fitness function

each chromosome in the population receives a fitness value. It biases the probability of the chromosome to reproduce. In our case the fitness value of a chromosome is the travel cost of its encoded solution.

4.2.3 Stop criterion

the algorithm ends after a fixed numbers of consecutive generations without any improvement, or after a fixed iterations number.

4.2.4 Unfeasibility management

the main difficulty in choosing a proper representation for highly constrained Combinatorial Optimization problems such as CVRP is dealing with the solutions in feasibilities generated during the evolutionary process. This problem is typically addressed by modifying the breeding operators, associating them with repair methods, providing penalties on unfeasible solutions in the fitness function, or discarding infeasible solutions when created. However, the use of penalty functions or rejecting strategies is inefficient for the CVRP, being the feasible schedules space very small, when compared to the possible schedules one. Therefore, the GA would waste most of its time producing and/or processing infeasible solutions. When it is too hard finding embedded constraints in the coding of individuals, repairing techniques are the best choice. A solution is unfeasible if some routes in the chromosome cause the vehicle to exceed its capacity. When this happens, in order to avoid unfeasible solutions, our algorithm provides to delete all the exceeding nodes and locates them into another vehicle, when possible. Otherwise a new vehicle is generated (see Algorithm 4 for more details).

4.2.5 Reproduction and Reinsertion Probability

Reproduction and Reinsertion Probability for each individual depends on a proportional fitness assignment such that each individual in the population receives a *ReProduction Probability* (*RPP*) and a *ReInsertion Probability* (*RIP*) depending both on their own objective value and the remaining individuals objective value:

$$RPP(j) = \frac{1/Fitness(j)}{\sum_{i=1}^{n} 1/Fitness(i)} \tag{3}$$

$$RIP(j) = \frac{Fitness(j)}{\sum_{i=1}^{n} Fitness(i)} \tag{4}$$

This probability is introduced in order to decrease the population homogeneity, by offering to the best individuals a chance to remain in the population, depending on single individuals fitness values diversity.

4.2.6 Genetic Operators

we consider two kinds of operators: crossover and mutation. Crossover operator is used to consolidate what has already been "learned" by the different individuals. Recombination of two solutions is the primary tool for achieving this goal. In fact, combinations of useful pieces of information from different individuals would result in better overall solutions, accelerating the search for the global optimal one. Mutation operator ensures that new aspects of a problem are considered. On the other hand, mutation operators introduces new information by means of random swapping of one or more genes.

The partially Mapped Crossover

The Partially Mapped Crossover (PMX) chooses a random cut point in both parents and considers the segments following the cut point as a partial mapping of the genes to be exchanged in the first parent to generate the offspring. It takes corresponding genes from the segment of both parents, locates both these genes in the first parent, and exchanges them. Hence, given two parents and a cut in a given position of the two parents (the cut position is the same for both the parents), the PMX operator builds an offspring according to the following algorithm:

1. Starting from the first gene of the first parent on the right of the cut, PMX checks the genes.

2. Let us assume to consider the genes in position i.

   - If the genes on position i of the two parents are different, in the offspring the gene of the first parent replaces the gene of the second one in position i. However, the $i-th$ gene of the first parent is surely present in another gene $j \neq i$ of the second one. Thus, in the offspring the gene in position j is replaced by the gene in position i in the second parent.

- If the genes in the position i are equal, their value is placed in position i in the offspring.

3. we move to the next gene on the right. If we are at the end of the solutions, we exit, otherwise we return in 2.

4. All the genes of the offspring on the left of the cut that have been not yet fixed assume the value of the second parent ones.

Figure 4 presents an example of the PMX. In this example we introduce a cut between the genes placed in the ninth and the tenth position. The resulting offspring corresponds to the second parent after the following changes: for each gene on the right side of the cut we replace the genes of the Parent2 with the Parent1 one. So, we replace the gene C_3 of Parent2 with the gene C_6 from Parent1. Moreover, gene C_3 is moved in position 4 (where C_6 appears in Parent2). Then, we consider the position 11. Having the same value for both the parents, it involves no action. When we consider the position 12, instead, we replace gene C_9 of Parent2 with the gene C_3 of the Parent1. At the same time, gene C_9 is moved to position 4. Finally, we consider position 13, that does not changes the offspring anymore.

Chain Mutation with threshold

The Chain Mutation builds chooses a stochastic number of genes k, builds an ordered list of them (the chain) and shifts the corresponding information (see Algorithm 2). In Figure 5 an example is presented. Let us assume we selected the 4 genes in position 4, 10, 6 and 0 for the Chain Mutation. In the new solution gene the value of gene 4 is placed in gene 10, the value of gene 10 in gene 6 and so on, up to the last gene in the chain (gene 0) whose value is placed in the position of the first gene of the chain (gene 4).

With a normal mutation probability ($5 - 6\%$) after some iterations the population members have the same objective fuction. This situation means that many of the chromosomes are very similar or, at worst, equal each others, but actually we are in a local minimum. To escape from local minima we introduce a *Mutation Probability with threshold*. This probability depends on the distance between the best and the worst solutions in the population.

$$\begin{aligned} MP = \; & (H_{MP} - L_{MP}) * \\ & max\left\{0, min\left\{1, \left\lfloor \frac{\Delta_p}{\alpha} \right\rfloor\right\}\right\} + L_{MP} \end{aligned} \tag{5}$$

where

$$\Delta_p = \frac{Fitness_{Worst} - Fitness_{Best}}{Fitness_{Best}} * 100$$

$$H_{MP} = High\ level\ of\ Mutation\ Probability$$

$$L_{MP} = Low\ level\ of\ Mutation\ Probability$$

4.2.7 Clones management

we define clones two individuals having the same fitness function. After a fixed number of iterations, we check if there exist clones in the population, in order to replace them with new individuals.

4.3 TS features

We implemented the TS as a procedure inner to the GA, which is applied to each offspring previously generated by our GA. Our TS explores a macro-neighborhood constituted by the fusion of two neighborhoods called the Add and Drop and 1-Swap neighborhood. At each iteration, on the Add and Drop neighborhood is applied a procedure that iterates it (namely the k-chain-moves procedure). The best solution of the macro-neighborhood is chosen as the Candidate Solution and this procedure is iterated until the stop criterion is not met. In the following the above mentioned components are presents.

4.3.1 Neighborhood

TS procedure starts by randomly choosing a target vehicle in order to generate its neighborhood. Our algorithm is able to combine two different kinds of neighborhoods: the Add and Drop and the 1-Swap neighborhood. The Add and Drop neighborhood consists in trying to put each vehicle's customer into each other vehicle. The 1-Swap considers each target vehicle's customer and attempts to swap it with another customer belonging to a different vehicle. These two neighborhoods consider only feasible solutions, and it means that solutions that lead vehicles to exceed their capacity are discarded. The whole neighborhood is generated and evaluated by the k-chain-moves.

4.3.2 k-chain-moves procedure

the k-chain-moves is a generalized method to increase the size and the effectiveness of a neighborhood without increasing its computational complexity. The basic idea, introduced by Crainic, Perboli and Tadei for the Multi-dimensional Bin Packing Problem ([22], [10]), is to build chains of moves of length k into a solution. Given a neighborhood of complexity $O(J)$, a chain of k moves can be trivially obtained in $O(J^k)$. For example, if a neighborhood considers all the pairs in a given set, a series of k moves can be obtained considering all the possible k-tuples. Unfortunately this approach can not be used for the growing of the computational effort needed to exploit the new neighborhood. In the k-chain-moves (see Algorithm 5) at each iteration the best solution given by the neighborhood at iteration i is directly applied as the current solution of the neighborhood of the iteration $i+1$. In such a way, the overall complexity becomes $O((k-1)J)$, where k is a constant. Consider, for example, the situation depicted in Figure 6. Let us assume to apply on the solution the Add and Drop neighborhood, to use V_1 as target vehicle and to obtain

that the best move is to load customer C_5 on vehicle V_2 (Figure 6-(a)). Now let us suppose to apply the k-chain-moves to the same neighborhood and to the same solution, setting the chain length equal to 2. At the end of the first iteration, we will obtain exactly the previous move as the best one (we move customer C_5 from vehicle V_1 to V_2). In the second iteration, the current solution is obtained from the current solution of the first iteration where customer C_5 is moved from vehicle V_1 to V_2 (see Figure 6-(b)). We apply back the Add and Drop using vehicle V_2 as target vehicle and forbidding to consider the vehicle V_1 anymore. Suppose now that the best move is to move customer C_{13} from vehicle V_2 to V_3. We obtained a chain of 2 moves and the resulting solution is depicted in Figure 6-(c).

4.3.3 Taboo List

any vertex removed from route r at iteration t is prevented from being reinserted in that vehicle until iteration $t + \vartheta$, where ϑ is the Taboo List size. The Taboo List size is dynamically set by the algorithm. It is increased of 1 unit if the best solution in the neighborhood is worst than the starting solution, or it is reduced at minimum size if the current solution is the best found so far. Moreover, a minimum and a maximum size for the TL is given.

4.3.4 Aspiration and stop criteria

the aspiration criterion allows the procedure to consider those moves whose objective value is better than the current best one, even if they are taboo moves. Both GA and TS components present the same stop criteria: they end after Q iterations without any improvement or after a fixed iterations number. Notice that TS is a inner function of the GA, so when TS ends the algorithm continues executing the GA procedure, while arresting the GA means stopping the whole algorithm.

5 Computational Results

In this section we present the results from the computational experiments. EVE-OPT has been implemented in C language and the tests have run on a Pentium-IV 3.0 GHz PC. Benchmark instances are built such that the vertices are defined by points in the plane and the cost for each edge (i, j) is the Euclidean distance between i and j. The evaluation is performed on two sets of Euclidean benchmark problems: the first contains 7 medium-sized classical problems with 50-199 customers proposed by Christofides et al. [6] and can be downloaded from the OR Library [1], the second contains 8 large-scale instances (200-480 customers) presented by Golden et al. [15]. To give an estimation of EVE-OPT's performances we present two series of results: in the first one we analyze the behavior of the different components of EVE-OPT, while in the second one EVE-OPT is compared with state-of-the-art algorithms.

5.1 EVE-OPT's components effect

EVE-OPT is characterized by different parameters that must be tuned. In particular, EVE-OPT is based on different sub-procedures that interact one with the other. In the following we present the effects on the global behavior of the hybridization between the GA and the TS, the Chain Mutation and the k-chain-moves algorithm.

5.1.1 EVE-OPT Vs GA and TS

aim of this test is to understand whether the joint approach of TS and GA is fruitful. The question is therefore whether EVE-OPT performs better than its two components taken separately. In Tables 1 and 2 the results of EVE-OPT, TS only and GA only are presented. The results have been obtained firstly using EVE-OPT with the GA and the TS hybridised (the TS stops after 50 non-improving iterations). Given the time needed by EVE-OPT to optimize each instance, the GA and the TS are applied separately to the same instance for the same time. In each table, the first column present the instance name, while the second is the number of customers in the instance. Columns 4, 6 and 8 give the objective function value obtained using the GA, the TS and the EVE-OPT respectively, while columns 5, 7 and 9 present the deviation of each algorithm from the best known solution, which value is reported in column 3. In Table 1 we can see how the TS outperforms the GA in the medium instances, while they produce similar results in the bigger ones (see Table 2). Our Memetic Algorithm takes profit from these two opposite behaviors, outperforming both TS and GA in both two sets of instances.

5.1.2 EVE-OPT with/without mutation threshold

fixing the mutation probability to a standard value ($5 - 6\%$), the population converges rather fast to a local minimum without any chance to escape. When increasing this value, instead, a strong stochastic component in introduced in the system, making it highly unpredictable. Our idea is to use a mutation probability depending from the distance between the best and the worst population fitness values. It allows us to exploit the stability of low mutation probability in eterogeneous populations and the strong stochastic component in the homogeneous ones, in order to explore new regions of the solution space when the population is composed by similar individuals. Computational results presented in Tables 3 and 4 are obtained setting a fixed mutation probability $L_{MP} = 6\%$ in the tests named *Without threshold* and using the dynamic mutation probability in the *With threshold* tests. In the second case, the parameters have been setted in the following way: $H_{MP} = 40\%$, $L_{MP} = 6\%$, $\alpha = 5\%$. More precisely, in each table, the first column present the instance name, while the second is the number of customers in the instance. The best known value for each instance is reported in column 3. Columns 4 and 6 report the objective function value obtained in the two tests, while columns 5 and 7 present the deviation from the best known solution. According to the those results, the mutation with a dynamic probability related to

the homogenous behaviour of the population leads to an improvement up to 1% of the overall result.

5.1.3 EVE-OPT with/without k-chain-moves

results listed in Tables 5 and 6 show the effectiveness of the k-chain-moves procedure. In each table, the first column present the instance name, while the second is the number of customers in the instance. The best known value for each instance is reported in column 3. Columns 4, 6 and 8 report the objective function value obtained setting the length of the chain of moves in the k-chain-moves procedure to 1, 2 and 3 respectively, while columns 5, 7 and 9 present the deviation from the best known solution. Notice that, when k is set to 1, this is equivalent to disable the k-chain-moves procedure. According to those result, the best values are obtained when k is 3. According to our tests, higher values of k do not improve the solution quality, increasing the execution time only. According to the results, setting $k = 3$ we obtain an improvement of the solution quality up to 5%.

5.2 EVE-OPT Vs State-of-art

In this section, we compare EVE-OPT with state-of-the-art algorithms. Tables 7 and 7 show the results of EVE-OPT compared with the results of GTS [30], VRTR [17], USTA [8], VLNS [11], Prins [23], Berger and Barkaoui [5] and AGES [19] in the normal and fast version. In each table, the first column present the instance name, while the second is the number of customers in the instance. The value of the objective function of the best known solution is reported in column 3. For each algorithm, we reported the value of objective function of the best solution, the gap from the best solution and the execution time. Given the different computers employed, the computational times are reported, even if they are hard to interpret.

According to our computational results, AGES has still the best results both in accuracy and computing time. On the other hand, AGES is the most complex algorithm to implement and one of the more complicated to introduce additional constraints.

EVE-OPT is able to achieve the same results of Prins on the medium-sized instances, while in the large-sized instances we obtain the same results on 5 instances and comparable results on the remaining 3. From the computational point of view, EVE-OPT is able to find its best solution with a reduction of the overall computational effort up to 2 times. Indeed, we have to consider the difference in computational power of the two machines used to run the tests of the two algorithms (Prins's computational times have been obtained using a 1 GHz Pentium-based PC, while EVE-OPT used a 3 GHz Pentium-based PC, see [2] for a comparison of the processors). Moreover, EVE-OPT is able to converge to less than 1% from the optimality in at most 2/3 of the total computation time (see for example Figure 7, where we report the value of the objective function in the biggest instance).

6 Conclusions

This paper describes EVE-OPT, a Memetic Algorithm for the CVRP combining Genetic Algorithms and Taboo Search. The ingredients of our GA are a natural representation of solutions and a recombination capable of transmitting meaningful characteristics from parents to children. Moreover, a new kind of mutation, the Chain Mutation, and a new effective dynamic mutation scheme are presented. Computational results suggest that all basic structures bring a considerable contribution to the solutions quality, and our algorithm generates results comparable with the State-of-Art. All its inner structures and procedures developed put particular attention on simplicity. Actually EVE-OT has been modified to be used in a real delivery problem. Moreover, we expect to apply it, in the near future, to more constrained academic and real problems.

References

[1] [Online]. Available: http://www.branchandcut.org/VRP/data/

[2] [Online]. Available: http://www.spec.org/cpu2000/results/

[3] G. Barbarosoglu and D. Ozgur, "A tabu search algorithm for the vehicle routing problem," *Comput. Oper. Res.*, vol. 26, no. 3, pp. 255–270, 1999.

[4] J. W. Barnes, M. Laguna, and F. Glover, *Intelligent Scheduling Systems*. Kluwer Academic, 1995.

[5] J. Berger and M. Barkaoui, "A new hybrid genetic algorithm for the capacitated vehicle routing problem," *Journal of the Operational Research Society*, vol. 54, pp. 1254–1262, 2004.

[6] N. Christofides, A. Mingozzi, and P. Toth, "The vehicle routing problem," *Combinatorial optimization*, pp. 315–338, 1979.

[7] G. Clarke and J. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Comput. Oper. Res.*, vol. 4, pp. 568–581, dic 1964.

[8] J.-F. Cordeau, G. Laporte, and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational Research Society*, vol. 52, pp. 928–936, 2001.

[9] J.-F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.-S. Sormany, "New heuristics for the vehicle routing problem," Les Cahiers du GERAD, HEC Montréal, Tech. Rep. G-2004-33, 2004.

[10] T. Crainic, G. Perboli, and R. Tadei, "Two-level tabu search for 3-d orthogonal bin packing," in *INFORMS Annual Meeting Denver 2004*. INFORMS, 2004.

[11] O. Ergun, J. B. Orlin, and A. Steele-Feldman, "Creating very large scale neighborhoods out of smaller ones by compounding moves: A study on the vehicle routing problem," 2003.

[12] M. Gendreau, A. Hertz, and G. Laporte, "A tabu search heuristic for the vehicle routing problem," *Manage. Sci.*, vol. 40, no. 10, pp. 1276–1290, 1994.

[13] H. Ghaziri, "Solving routing problems by a self-organizing map," pp. 829–834.

[14] ——, "Supervision in the self-organizing feature map: Application to the vehicle routing problem," pp. 651–660.

[15] B. L. Golden, E. A. Wasil, J. Kelly, and I. M. Chao, "The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results," *Fleet management and logistics*, pp. 33–56, 1998.

[16] B. Golden, E. Wasil, J. Kelly, and I. Chao, "Metaheuristics in vehicle routing," *Fleet Management and Logistics*, pp. 33–56, 1998.

[17] F. Li, B. Golden, and E. Wasil, "Very large-scale vehicle routing: New test problems, algorithms, and results," *Comput. Oper. Res. Forthcoming*, 2004.

[18] Y. Matsuyama, "Self-organization via competition, cooperation and categorization applied to extended vehicle routing problems," in *Proceedings of the International Joint Conference on Neural Networks*, Seatle, WA, 1991, pp. 385–390.

[19] D. Mester and O. Bräysy, "Active guided evolution strategies for the large scale vehicle routing problems with time windows," *Comput. Oper. Res. Forthcoming*, 2004.

[20] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[21] I. H. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Ann. Oper. Res.*, vol. 41, no. 1-4, pp. 421–451, 1993.

[22] G. Perboli, "Bounds and heuristics for the packing problems," Ph.D. dissertation, 2002.

[23] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," *Comput. Oper. Res.*, vol. 31, pp. 1985–2002, 2004.

[24] C. Rego, "A subpath ejection method for the vehicle routing problem," *Management Science*, vol. 44, pp. 1447–1459, 1998.

[25] C. Rego and C. Roucairol, "A parallel tabu search algorithm using ejection chains for the vehicle routing problem," pp. 661–675, 1996.

[26] M. Reimann, K. Doerner, and R. F. Hartl, "D-ants: savings based ants divide and conquer the vehicle routing problem," *Comput. Oper. Res.*, vol. 31, no. 4, pp. 563–591, 2004.

[27] Y. Rochat and E. Taillard, "Probabilistic diversification and intensification in local search for vehicle routing," *Journal of Heuristics*, pp. 147–167, 1995.

[28] M. Schumann and R. Retzko, "Self–organizing maps for vehicle routing problems – minimizing an explicit cost function," in *Fogelman-Soulie, F. (ed.), Proceedings of the International Conference on Artificial Neural Networks*, Paris, France, 1995, pp. 401–406.

[29] C.-D. Tarantilis and C. Kiranoudis, "Bone route: An adaptive memory-based method for effective fleet management," *Ann. Oper.Res.*, no. 115, pp. 227–241, 2002.

[30] P. Toth and D. Vigo, "The granular tabu search and its application to the vehicle routing problem," *Journal on Computing*, no. 15, pp. 333–346, 2003.

[31] P. Toth and D. Vigo, Eds., *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 2001.

[32] J. Xu and J. Kelly, "A network flow-based tabu search heuristic for the vehicle routing problem," *Transportation Science*, vol. 30, pp. 379–393, 1996.

Algorithm 1 Generating Swaps Set of Genes S

$v(i)$: path containing customer i
$k = rand(n)$: number of genes to swap
$j = rand(n)$: index of swap current gene
$S = \{\varnothing\}$: Set of swap genes

$S = S \cup j$
for $i = 2$ **to** k **do**
  $j_{prev} = j$
  while $v(j) = v(j_{prev})$ **do**
    $j = rand(n)$
  end while
  $S = S \cup j$
end for
return S

Algorithm 2 Chain Mutation

Input C: target chromosome
Input S: set of swap genes generated according to Algorithm 1

$prev = C(S(1))$
for $i = 2$ **to** k **do**
  $current = C(S(i))$
  $C(S(i)) = prev$
  $prev = current$
end for
$C(S(1)) = prev$
return C

Algorithm 3 Taboo Search

Input SS: Starting Solution
$BS = SS$: Best Solution
$CS = SS$: Current Solution
$TL = \{\oslash\}$: Taboo List
while Stop criterion not met **do**
  Generate the CS neighborhood
  S: best solution in CS neighborhood
  if $O.F.(S) \succ O.F.(BS)$ **then**
    $BS = S$
  end if
  Update TL
  Update CS
end while

Algorithm 4 Feasibility procedure

Input $\bar{v}$: target vehicle
Input G: set of unfeasible genes
Input M: set of vehicles
$GSet$: set of possible moves for gene g

for all $g \in G$ **do**
  $GSet = \{\oslash\}$
  for $j = 1$ **to** $|M|$ **do**
    if $move(i, j)$ is feasible **then**
      $GSet = GSet \cup \{move(i, j)\}$
    end if
  end for
  Move i in the best move $\in GSet$
end for

Algorithm 5 k-chain-moves procedure

Input K: length of the chain to build
Input CS: current solution
$N^1(i)$: Add and Drop neighborhood of solution i
$N^2(i)$: 1-Swap neighborhood of solution i
$N(i)$: whole neighborhood of solution i
BS: best solution
BSN: neighborhood best solution

$N^1(CS) = N^2(CS) = N(CS) = \{\varnothing\}$
$BS = BSN = NULL$

for $i = 1$ **to** K **do**
  Generate $N^1(CS)$
  if $i = 1$ **then**
    Generate $N^2(CS)$
  end if
  $N(CS) = N^1(CS) \cup N^2(CS)$
  Evaluate $N(CS)$ and select BSN
  if BSN is better than BS **then**
    $BS = BSN$
  end if
  $CS = BSN$
  $N^2(CS) = \{\varnothing\}$
end for
Optimize all paths with a Branch and Bound procedure
Update the Taboo List
return BS

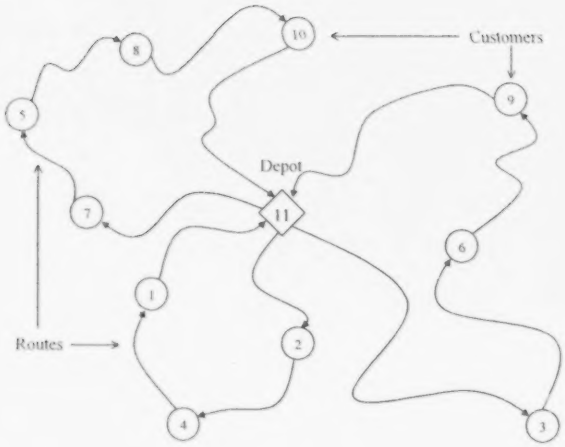

Figure 1: Example of a VRP solution

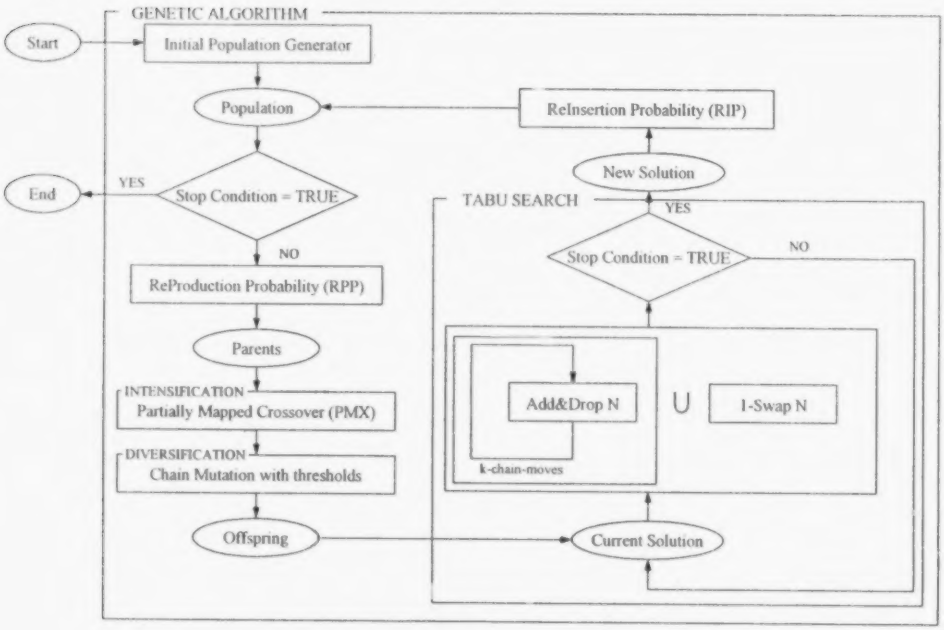

Figure 2: EVE-OPT schema

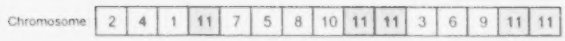

Figure 3: Example of a chromosome

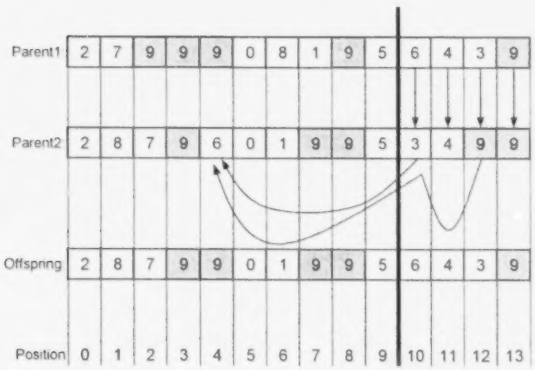

Figure 4: Example of the Partially Mapped Crossover

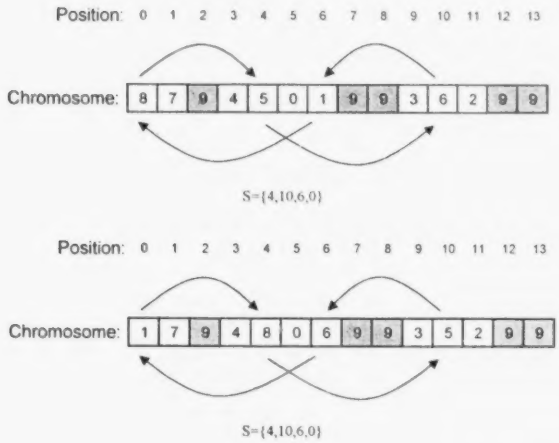

Figure 5: Chromosome before and after the Chain Mutation

Figure 3: Example of a chromosome

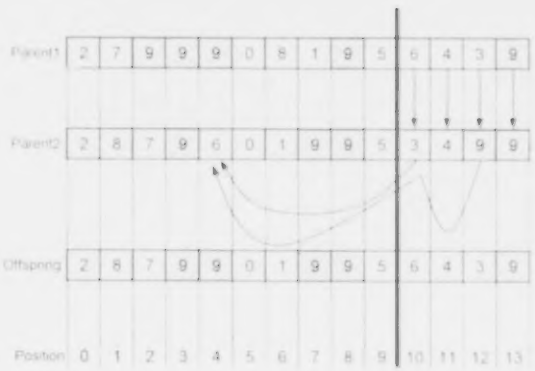

Figure 4: Example of the Partially Mapped Crossover

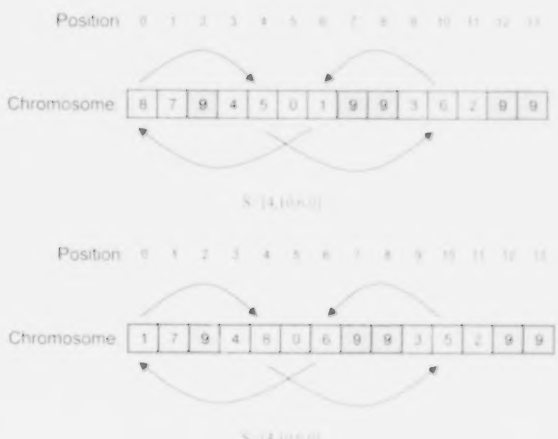

Figure 5: Chromosome before and after the Cham Mutation

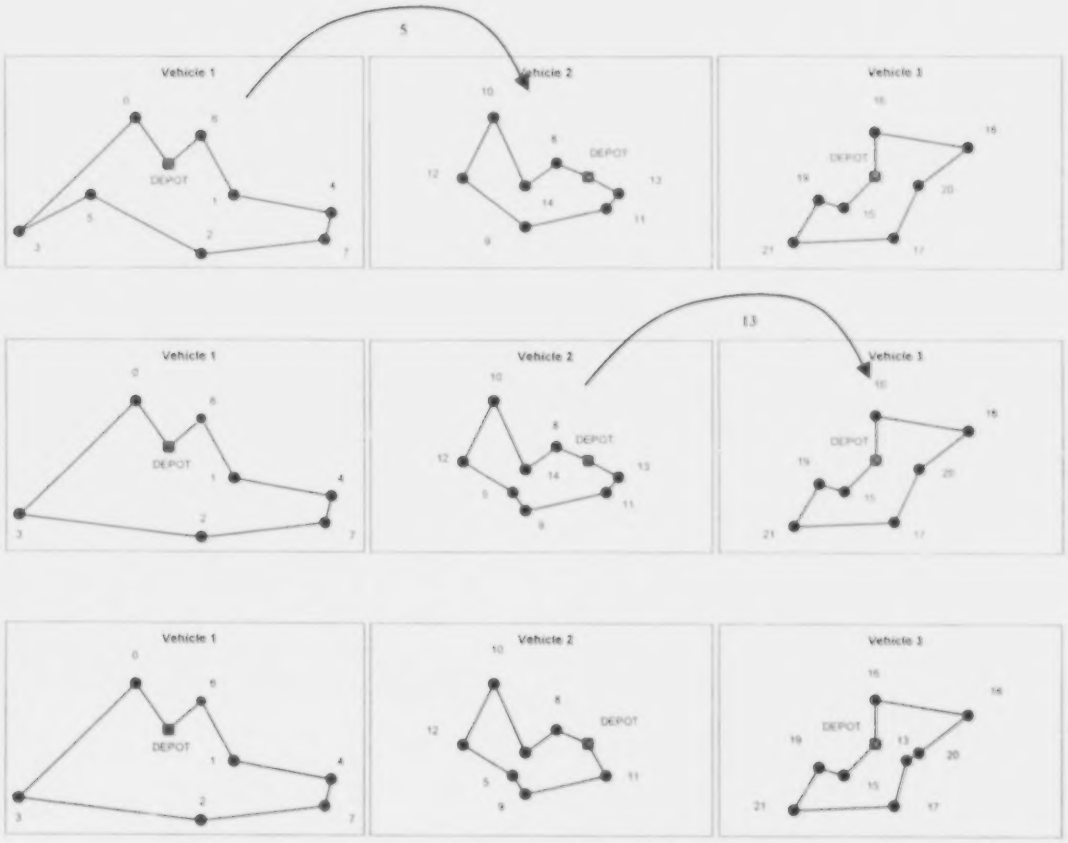

Figure 6: Example of k-chain-moves

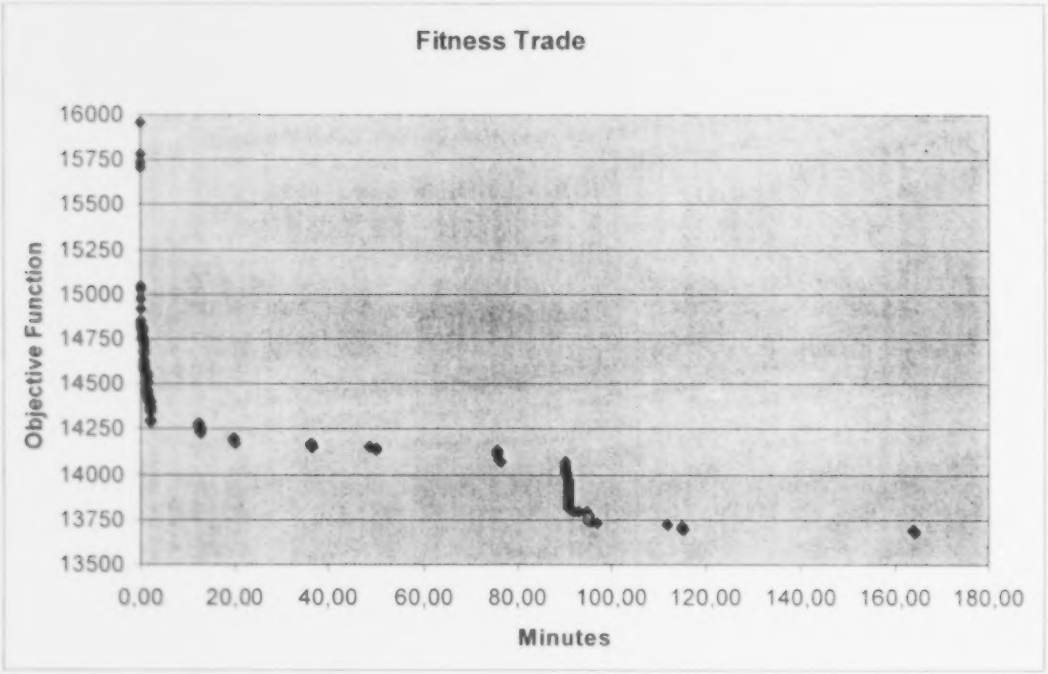

Figure 7: Convergence to the optimality of the objective function in the Instance 4 by Golden et al.

Instance		Best	GA		TS		EVE-OPT	
			Value	%	Value	%	**Value**	**%**
1	50	524.61	531.32	1.28%	524.61	0.00%	**524,61**	**0,00%**
2	75	835.26	842.93	0.92%	835.26	0.00%	**835,26**	**0,00%**
3	100	826.14	845.65	2.36%	827.04	0.11%	**826,14**	**0,00%**
4	150	1028.42	1065.67	3.62%	1043.29	1.45%	**1030,42**	**0,19%**
5	199	1291.29	1332.34	3.18%	1321.11	2.31%	**1300,23**	**0,69%**
11	120	1042.11	1087.64	4.37%	1079.98	3.63%	**1042,11**	**0,00%**
12	100	819.56	837.82	2.23%	819.56	0.00%	**819,56**	**0,00%**
Average				2.57%		1.07%		**0,13%**

Table 1: Comparison between EVE-OPT, TS e GA: Christofides at al. instances

Instance		Best	GA		TS		EVE-OPT	
			Value	%	Value	%	**Value**	**%**
1	240	5627.54	5959.49	5.90%	5915.00	5.11%	**5642,02**	**0,26%**
2	320	8447.92	8998.61	6.52%	8950.25	5.95%	**8447,92**	**0,00%**
3	400	11036.22	12501.83	13.28%	12664.31	14.75%	**11121,98**	**0,78%**
4	480	13624.52	14982.54	9.97%	15133.37	11.07%	**13673,58**	**0,36%**
5	200	6460.98	6822.67	5.60%	6635.47	2.70%	**6460,98**	**0,00%**
6	280	8412.8	8635.61	2.65%	8602.89	2.26%	**8412,80**	**0,00%**
7	360	10195.56	11178.69	9.64%	11152.29	9.38%	**10215,19**	**0,19%**
8	440	11663.55	12948.45	11.02%	12836.94	10.06%	**11828,36**	**1,41%**
Average				8.07%		7.66%		**0,38%**

Table 2: Comparison between EVE-OPT, TS e GA: Golden at al. instances

			GA		TS		EVE-OPT	
Instance		Best	Value	%	Value	%	Value	%
1	50	524,61	531,32	1,28%	524,61	0,00%	**524,61**	**0,00%**
2	75	835,26	842,93	0,92%	835,26	0,00%	**835,26**	**0,00%**
3	100	826,14	845,65	2,36%	827,04	0,11%	**826,14**	**0,00%**
4	150	1028,42	1065,67	3,62%	1043,29	1,45%	**1030,42**	**0,19%**
5	199	1291,29	1332,34	3,18%	1321,11	2,31%	**1300,23**	**0,69%**
11	120	1042,11	1087,64	4,37%	1079,98	3,63%	**1042,11**	**0,00%**
12	100	819,56	837,82	2,23%	819,56	0,00%	**819,56**	**0,00%**
Average				2,57%		1,07%		**0,13%**

Table 1: Comparison between EVE-OPT, TS e GA: Christofides at al. instances

			GA		TS		EVE-OPT	
Instance		Best	Value	%	Value	%	Value	%
1	240	5627,54	5959,49	5,90%	5915,00	5,11%	**5642,02**	**0,26%**
2	320	8447,92	8998,61	6,52%	8950,25	5,95%	**8447,92**	**0,00%**
3	400	11036,22	12501,83	13,28%	12664,31	14,75%	**11121,98**	**0,78%**
4	480	13624,52	14982,54	9,97%	15133,37	11,07%	**13673,58**	**0,36%**
5	200	6460,98	6822,67	5,60%	6635,47	2,70%	**6460,98**	**0,00%**
6	280	8412,8	8635,61	2,65%	8602,89	2,26%	**8412,80**	**0,00%**
7	360	10195,56	11178,69	9,64%	11152,29	9,38%	**10215,19**	**0,19%**
8	440	11663,55	12948,45	11,02%	12836,94	10,06%	**11828,36**	**1,41%**
Average				8,07%		7,66%		**0,38%**

Table 2: Comparison between EVE-OPT, TS e GA: Golden at al. instances

Instance		Best	Without threshold		With threshold	
			Value	%	Value	%
1	50	524,61	524,61	0,00%	524,61	0,00%
2	75	835,26	835,26	0,00%	835,26	0,00%
3	100	826,14	826,14	0,00%	826,14	0,00%
4	150	1028,42	1033,20	0,46%	1030,42	0,19%
5	199	1291,29	1302,95	0,90%	1300,23	0,69%
11	120	1042,11	1044,71	0,25%	1042,11	0,00%
12	100	819,56	819,56	0,00%	819,56	0,00%
Average				0,23%		0,13%

Table 3: Comparison between EVE-OPT with and without Threshold Probability: Christofides at al. instances

Instance		Best	Without threshold		With threshold	
			Value	%	Value	%
1	240	5627,54	5662,10	0,61%	5642,02	0,26%
2	320	8447,92	8505,49	0,68%	8447,92	0,00%
3	400	11036,22	11121,71	0,77%	11121,98	0,78%
4	480	13624,52	13703,56	0,58%	13673,58	0,36%
5	200	6460,98	6518,34	0,89%	6460,98	0,00%
6	280	8412,80	8420,37	0,09%	8412,80	0,00%
7	360	10195,56	10213,58	0,18%	10215,19	0,19%
8	440	11663,55	11895,64	1,99%	11828,36	1,41%
Average				0,72%		0,38%

Table 4: Comparison between EVE-OPT with and without Threshold Probability: Golden at al. instances

			K=1		K=2		K=3	
Instance		Best	Value	%	Value	%	Value	%
1	50	524,61	524,61	0,00%	524,61	0,00%	524,61	0,00%
2	75	835,26	835,26	0,00%	835,26	0,00%	835,26	0,00%
3	100	826,14	827,14	0,12%	826,14	0,00%	826,14	0,00%
4	150	1028,42	1035,52	0,69%	1032,42	0,39%	1030,42	0,19%
5	199	1291,29	1329,19	2,94%	1301,82	0,82%	1300,23	0,69%
11	120	1042,11	1047,11	0,48%	1042,11	0,00%	1042,11	0,00%
12	100	819,56	827,09	0,92%	819,56	0,00%	819,56	0,00%
Average				0,74%		0,17%		0,13%

Table 5: Results with different values of k in the k-chain-moves procedure: Christofides at al. instances

			K=1		K=2		K=3	
Instance		Best	Value	%	Value	%	Value	%
1	240	5627,54	5737,02	1,95%	5677,22	0,88%	5642,02	0,26%
2	320	8447,92	8651,92	2,41%	8505,19	0,68%	8447,92	0,00%
3	400	11036,22	11430,88	3,58%	11446,92	3,72%	11121,98	0,78%
4	480	13624,52	14283,29	4,84%	13813,27	1,39%	13673,58	0,36%
5	200	6460,98	6591,63	2,02%	6509,63	0,75%	6460,98	0,00%
6	280	8412,80	8612,48	2,37%	8491,48	0,94%	8412,80	0,00%
7	360	10195,56	10561,28	3,59%	10493,45	2,92%	10215,19	0,19%
8	440	11663,55	12175,26	4,39%	11922,04	2,22%	11828,36	1,41%
Average				3,14%		1,69%		0,38%

Table 6: Results with different values of k in the k-chain-moves procedure: Golden at al. instances

Table 7: EVE-OPT Vs State-Of-Art: Christofides at al. instances

			GTS Toth and Vigo (2003)			Li, Golden and Wasil (2004)			USTA Cordeau et al. (2001)			VLNS Ergun et al. (2003)			Prins (2004)		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes
1	50	524,61	524,61	0,00%	0,81	524,61	0,00%		524,61	0,00%	2,32	524,61	0,00%	23,13	524,61	0,00%	0,01
2	75	835,26	838,60	0,40%	2,21	836,18	0,11%		835,28	0,00%	14,78	835,43	0,02%	33,93	835,26	0,00%	0,77
3	100	826,14	828,56	0,29%	2,39	827,39	0,15%		826,14	0,00%	11,67	827,46	0,16%	21,30	826,14	0,00%	0,46
4	150	1028,42	1033,21	0,47%	4,51	1045,36	1,65%		1032,68	0,41%	26,66	1036,24	0,76%	24,45	1031,63	0,31%	5,50
5	199	1291,29	1318,25	2,09%	7,50	1303,47	0,94%		1315,76	1,90%	57,68	1307,33	1,24%	57,25	1300,23	0,69%	19,11
11	120	1042,11	1042,87	0,07%	0,86	1042,11	0,00%		1073,47	3,01%	11,67	1042,11	0,00%	69,13	1042,11	0,00%	0,30
12	100	819,56	819,56	0,00%	2,75	819,56	0,00%		819,56	0,00%	9,02	819,56	0,00%	5,98	819,56	0,00%	0,05
Average				0,47%	3,00		0,41%	0,00		0,76%	19,11		0,31%	33,60		0,14%	3,74

			AGES best Mester and Bräysy (2004)			AGES fast Mester and Bräysy (2004)			Berger and Barkaoui (2004)			**EVE-OPT**		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	**Value**	**%**	**Minutes**
1	50	524,61	524,61	0,00%	0,01	524,61	0,00%		524,61	0,00%	2,00	**524,61**	**0,00%**	**0,94**
2	75	835,26	835,26	0,00%	0,26	835,26	0,00%		835,26	0,00%	14,33	**835,26**	**0,00%**	**7,17**
3	100	826,14	826,14	0,00%	0,05	826,14	0,00%		827,39	0,15%	27,90	**826,14**	**0,00%**	**6,92**
4	150	1028,42	1028,42	0,00%	0,47	1028,42	0,00%		1036,16	0,75%	48,98	**1030,42**	**0,19%**	**30,26**
5	199	1291,29	1291,29	0,00%	101,93	1294,25	0,23%		1324,06	2,54%	55,41	**1300,23**	**0,69%**	**11,51**
11	120	1042,11	1042,11	0,00%	0,05	1042,11	0,00%		1043,11	0,10%	22,43	**1042,11**	**0,00%**	**4,64**
12	100	819,56	819,56	0,00%	0,01	819,56	0,00%		819,56	0,00%	7,21	**819,56**	**0,00%**	**20,83**
Average				0,00%	14,68		0,03%			0,51%	25,47		**0,13%**	**11,75**

Table 7: EVE-OPT vs State-Of-Art Christofides et al. instances

Instance		Best	GTS Toth and Vigo (2003)			Li, Golden and Wasil (2004)			USTA Cordeau et al. (2001)			VLNS Ergun et al. (2003)			Prins (2004)		
			Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes
1	50	524.61	524.61	0.00%	0.81	524.61	0.00%		524.61	0.00%	2.32	524.61	0.00%	23.13	524.61	0.00%	0.01
2	75	835.26	838.60	0.40%	2.21	836.18	0.11%		835.28	0.00%	14.78	835.43	0.02%	33.93	835.26	0.00%	0.77
3	100	826.14	828.56	0.29%	2.39	827.39	0.15%		826.14	0.00%	11.67	827.46	0.16%	21.30	826.14	0.00%	0.46
4	150	1028.42	1033.21	0.47%	4.51	1045.36	1.65%		1032.68	0.41%	26.66	1036.24	0.76%	24.45	1031.63	0.31%	5.50
5	199	1291.29	1318.25	2.09%	7.50	1303.47	0.94%		1315.76	1.90%	57.68	1307.33	1.24%	57.25	1300.23	0.69%	19.11
11	120	1042.11	1042.87	0.07%	0.86	1042.11	0.00%		1073.47	3.01%	11.67	1042.11	0.00%	69.13	1042.11	0.00%	0.30
12	100	819.56	819.56	0.00%	2.75	819.56	0.00%		819.56	0.00%	9.02	819.56	0.00%	5.98	819.56	0.00%	0.05
Average				0.47%	3.00		0.41%	0.00		0.76%	19.11		0.31%	33.60		0.14%	3.74

Instance		Best	AGES best Mester and Braysy (2004)			AGES fast Mester and Braysy (2004)			Berger and Barkaoui (2004)			**EVE-OPT**		
			Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	**Value**	**%**	**Minutes**
1	50	524.61	524.61	0.00%	0.01	524.61	0.00%		524.61	0.00%	2.00	**524,61**	**0,00%**	**0,94**
2	75	835.26	835.26	0.00%	0.26	835.26	0.00%		835.26	0.00%	14.33	**835,26**	**0,00%**	**7,17**
3	100	826.14	826.14	0.00%	0.05	826.14	0.00%		827.39	0.15%	27.90	**826,14**	**0,00%**	**6,92**
4	150	1028.42	1028.42	0.00%	0.47	1028.42	0.00%		1036.16	0.75%	48.98	**1030,42**	**0,19%**	**30,26**
5	199	1291.29	1291.29	0.00%	101.93	1294.25	0.23%		1324.06	2.54%	55.41	**1300,23**	**0,69%**	**11,51**
11	120	1042.11	1042.11	0.00%	0.05	1042.11	0.00%		1043.11	0.10%	22.43	**1042,11**	**0,00%**	**4,64**
12	100	819.56	819.56	0.00%	0.01	819.56	0.00%		819.56	0.00%	7.21	**819,56**	**0,00%**	**20,83**
Average				0.00%	14.68		0.03%			0.51%	25.47		**0,13%**	**11,75**

			GTS Toth and Vigo (2003)			Li, Golden and Wasil (2004)			USTA Cordeau et al. (2001)			VLNS Ergun et al. (2003)			Prins (2004)		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes
1	240	5627,54	5736,15	1,93%	4,98	5666,42	0,69%		5681,97	0,97%	10,29	5741,79	2,03%	134,95	5646,63	0,34%	32,42
2	320	8447,92	8553,03	1,24%	8,28	8469,32	0,25%		8657,36	2,48%	35,39	8917,41	5,56%	150,83	8447,92	0,00%	77,92
3	400	11036,22	11402,75	3,32%	12,94	11145,80	0,99%		11037,40	0,01%	55,39	12106,64	9,70%	15,67	11036,22	0,00%	120,83
4	480	13624,52	14910,62	9,44%	15,13	13758,08	0,98%		13740,60	0,85%	83,19	15316,69	12,42%	106,50	13624,52	0,00%	187,60
5	200	6460,98	6697,53	3,66%	2,38	6478,09	0,26%		6756,44	4,57%	5,13	6570,28	1,69%	15,50	6460,98	0,00%	1,04
6	280	8412,80	8963,32	6,54%	4,65	8539,61	1,51%		8537,17	1,48%	18,64	8836,25	5,03%	81,98	8412,80	0,00%	9,97
7	360	10195,56	10547,44	3,45%	11,66	10289,72	0,92%		10267,40	0,70%	25,60	11116,68	9,03%	85,00	10195,59	0,00%	39,05
8	440	11663,55	12036,24	3,20%	11,08	11920,52	2,20%		11869,50	1,77%	71,44	12634,17	8,32%	33,95	11828,78	1,42%	88,30
Average				4,10%	8,89		0,98%			1,60%	38,13		6,72%	78,05		0,22%	69,64

			AGES best Mester and Bräysy (2004)			AGES fast Mester and Bräysy (2004)			Berger and Barkaoui (2004)			**EVE-OPT**		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	**Value**	**%**	**Minutes**
1	240	5627,54	5627,54	0,00%	8,73	5644,00	0,29%	0,70	5644,02	0,29%	62,52	**5642,02**	**0,26%**	**26,72**
2	320	8447,92	8447,92	0,00%	46,66	8468,00	0,24%	0,20	8449,12	0,01%	57,67	**8447,92**	**0,00%**	**27,58**
3	400	11036,22	11036,22	0,00%	40,55	11146,00	0,99%	0,70	11036,22	0,00%	21,92	**11121,98**	**0,78%**	**99,81**
4	480	13624,52	13624,52	0,00%	470,00	13704,52	0,59%	2,50	13699,11	0,55%	119,12	**13673,58**	**0,36%**	**188,94**
5	200	6460,98	6460,98	0,00%	0,17	6466,00	0,08%	0,50	6460,98	0,00%	0,87	**6460,98**	**0,00%**	**9,80**
6	280	8412,80	8412,88	0,00%	75,22	8539,61	1,51%	0,10	8412,90	0,00%	5,72	**8412,80**	**0,00%**	**31,34**
7	360	10195,56	10195,56	0,00%	0,55	10240,42	0,44%	0,85	10195,59	0,00%	14,03	**10215,19**	**0,19%**	**45,13**
8	440	11663,55	11663,55	0,00%	34,30	11918,75	2,19%	0,27	11828,78	1,42%	35,30	**11828,36**	**1,41%**	**72,90**
Average				0,00%	84,52		0,79%	0,73		0,28%	39,64		**0,38%**	**62,78**

Table 8: EVE-OPT Vs State-of-the-Art: Golden et al. instances

			GTS Toth and Vigo (2003)			Li, Golden and Wasil (2004)			USTA Cordeau et al. (2001)			VLNS Ergun et al. (2003)			Prins (2004)		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes
1	240	5627,54	5736,15	1,93%	4,98	5666,42	0,69%		5681,97	0,97%	10,29	5741,79	2,03%	134,95	5646,63	0,34%	32,42
2	320	8447,92	8553,03	1,24%	8,28	8469,32	0,25%		8657,36	2,48%	35,39	8917,41	5,56%	150,83	8447,92	0,00%	77,92
3	400	11036,22	11402,75	3,32%	12,94	11145,80	0,99%		11037,40	0,01%	55,39	12106,64	9,70%	15,67	11036,22	0,00%	120,83
4	480	13624,52	14910,62	9,44%	15,13	13758,08	0,98%		13740,60	0,85%	83,19	15316,69	12,42%	106,50	13624,52	0,00%	187,60
5	200	6460,98	6697,53	3,66%	2,38	6478,09	0,26%		6756,44	4,57%	5,13	6570,28	1,69%	15,50	6460,98	0,00%	1,04
6	280	8412,80	8963,32	6,54%	4,65	8539,61	1,51%		8537,17	1,48%	18,64	8836,25	5,03%	81,98	8412,80	0,00%	9,97
7	360	10195,56	10547,44	3,45%	11,66	10289,72	0,92%		10267,40	0,70%	25,60	11116,68	9,03%	85,00	10195,59	0,00%	39,05
8	440	11663,55	12036,24	3,20%	11,08	11920,52	2,20%		11869,50	1,77%	71,44	12634,17	8,32%	33,95	11828,78	1,42%	88,30
Average				4,10%	8,89		0,98%			1,60%	38,13		6,72%	78,05		0,22%	69,64

			AGES best Mester and Bräysy (2004)			AGES fast Mester and Bräysy (2004)			Berger and Barkaoui (2004)			**EVE-OPT**		
Instance		Best	Value	%	Minutes	Value	%	Minutes	Value	%	Minutes	**Value**	**%**	**Minutes**
1	240	5627,54	5627,54	0,00%	8,73	5644,00	0,29%	0,70	5644,02	0,29%	62,52	**5642,02**	**0,26%**	**26,72**
2	320	8447,92	8447,92	0,00%	46,66	8468,00	0,24%	0,20	8449,12	0,01%	57,67	**8447,92**	**0,00%**	**27,58**
3	400	11036,22	11036,22	0,00%	40,55	11146,00	0,99%	0,70	11036,22	0,00%	21,92	**11121,98**	**0,78%**	**99,81**
4	480	13624,52	13624,52	0,00%	470,00	13704,52	0,59%	2,50	13699,11	0,55%	119,12	**13673,58**	**0,36%**	**188,94**
5	200	6460,98	6460,98	0,00%	0,17	6466,00	0,08%	0,50	6460,98	0,00%	0,87	**6460,98**	**0,00%**	**9,80**
6	280	8412,80	8412,88	0,00%	75,22	8539,61	1,51%	0,10	8412,90	0,00%	5,72	**8412,80**	**0,00%**	**31,34**
7	360	10195,56	10195,56	0,00%	0,55	10240,42	0,44%	0,85	10195,59	0,00%	14,03	**10215,19**	**0,19%**	**45,13**
8	440	11663,55	11663,55	0,00%	34,30	11918,75	2,19%	0,27	11828,78	1,42%	35,30	**11828,36**	**1,41%**	**72,90**
Average				0,00%	84,52		0,79%	0,73		0,28%	39,64		**0,38%**	**62,78**

Table 8: EVE-OPT Vs State-of-the-Art: Golden at al. instances